Operační systém UNIX vznikl okolo roku
 :r1 1950
 :r2 1960
 :r3 1970
 :r4 1980
 :r5 1990
:r3 ok
--
První verze UNIXu byla naprogramována pro počítač
 :r1 IBM PC
 :r2 PDP
 :r3 IBM 370
 :r4 VAX
 :r5 Sun
:r2 ok
--
První verze UNIXu byla vytvořena v
 :r1 Bell Laboratories
 :r2 University of Berkeley
 :r3 Microsoftu
 :r4 Novellu
 :r5 USL (UNIX System Laboratories)
:r1 ok
--
Co vzniklo dříve? MS-DOS (předchůdce MS-Windows) nebo UNIX?
 :r1 MS-DOS
 :r2 UNIX
 :r3 Obojí je zhruba stejně staré.
:r2 ok
--
Kdo jsou autoři prvních verzí UNIXu?
 :r1 Steeve Jobs, Tim O'Reilly, Andrew Tanenbaum
 :r2 Larry Ellison, Scott McNealy, Michael Tiemann
 :r3 Brian Kernighan, Dennis Ritchie, Ken Thompson
 :r4 Eric Raymond, Richard Stallman, Linus Torvalds
:r3 ok
--
Operační systém Linux vzniká od roku
 :r1 1951
 :r2 1961
 :r3 1971
 :r4 1981
 :r5 1991
:r5 ok
--
Záznamy o uživatelích, kteří mají právo přistupovat k systému, jsou uloženy v souboru:
 :r1 /etc/group
 :r2 /etc/users
 :r3 /etc/passwd
 :r4 /etc/shadow
 :r5 /etc/motd
:r3 ok
--
Hesla uživatelů v otevřeném tvaru jsou uložena v uživatelům nepřístupném souboru:
 :r1 /etc/passwd
 :r2 /etc/passwords
 :r3 /etc/shadow
 :r4 nejsou uložena
:r4 ok
--
Soubor /etc/passwd v aktuálních verzích systému
 :r1 obsahuje zašifrovaná hesla uživatelů
 :r2 neobsahuje hesla uživatelů
 :r3 nikdy, ani v minulosti neobsahoval hesla uživatelů
 :r4 obsahuje nezašifrovaná hesla uživatelů
:r2 ok
--
Účet uživatele neobsahuje
 :r1 primární skupinu
 :r2 uživatelské jméno
 :r3 UID uživatele
 :r4 UČO uživatele
 :r5 žádná z ostatních odpovědí není správná
:r4 ok
--
Hesla uživatelů v zašifrovaném tvaru jsou uložena v uživatelům nepřístupném souboru:
 :r1 /etc/passwd
 :r2 /etc/shadow
 :r3 /etc/passwords
 :r4 nejsou uložena
:r2 ok
--
Uživatel zapomněl svoje heslo. Jak jej nejsnáze získá?
 :r1 uživatel použije příkaz decryptpas
 :r2 uživatel požádá superuživatele o provedení příkazu decryptpas
 :r3 uživatel požádá superuživatele o získání svého hesla ze souboru /etc/shadow
 :r4 žádná jiná odpověď není správná
:r4 ok
--
Uživatelské jméno se nazývá:
 :r1 login
 :r2 loggin
 :r3 user name
 :r4 uid
 :r5 user
:r1 ok
--
Hodnota UID u uživatele root je
 :r1 A
 :r2 root
 :r3 0
 :r4 1
 :r5 99999
:r3 ok
--
Uživatel je členem
 :r1 nemusí být členem žádné skupiny
 :r2 je členem alespoň jedné skupiny
 :r3 je členem alespoň dvou skupin
:r2 ok
--
Seznam skupin uživatelů v systému je uložen v souboru
 :r1 /etc/team
 :r2 /etc/groups
 :r3 /etc/group
 :r4 /etc/teams
 :r5 /etc/cluster
:r3 ok
--
Soubor /etc/passwd musí být
 :r1 čitelný pouze superuživateli
 :r2 nesmí být čitelný nikomu
 :r3 čitelný pro všechny
 :r4 čitelný pouze skupině root
 :r5 žádná jiná odpověď není správná
:r3 ok
--
Soubor /etc/shadow musí být
 :r1 čitelný pouze superuživateli
 :r2 nesmí být čitelný nikomu
 :r3 čitelný pro všechny
 :r4 žádná jiná odpověď není správná
:r1 ok
--
Každé uživatelské jméno v souboru /etc/passwd musí být definováno
 :r1 alespoň jednou
 :r2 právě jednou
 :r3 nemusí být definováno vůbec
 :r4 právě dvakrát
 :r5 alespoň dvakrát
:r2 ok
--
Každé uživatelské jméno v souboru /etc/group musí být použito
 :r1 alespoň jednou
 :r2 právě jednou
 :r3 nemusí být uvedeno vůbec
 :r4 právě dvakrát
 :r5 alespoň dvakrát
:r3 ok
--
Interpret příkazů a domácí adresář, který se spustí a nastaví po přihlášení uživatele, je popsán
 :r1 v souboru .profile
 :r2 v účtu uživatele
 :r3 stavem při posledním odhlášení
 :r4 v souboru /etc/userdef
 :r5 v UID záznamu
:r2 ok
--
Čas v unixu je uložen (např. čas změny obsahu souboru)
 :r1 ve tvaru yyyymmddhhmiss
 :r2 ve tvaru ddmmyyyyhhmiss
 :r3 v počtu hodin od začátku epochy
 :r4 v počtu minut od začátku epochy
 :r5 v počtu sekund od začátku epochy
:r5 ok
--
Epocha v unixu začala
 :r1 1. 1. 1970 00:00
 :r2 31. 12. 1970 24:00
 :r4 1. 1. 1980 00:00
 :r5 31. 12. 1980 24:00
:r1 ok
--
Počet údajů na řádku v souboru /etc/passwd je (některý může být nevýznamný) alespoň
 :r1 4
 :r2 5
 :r3 6
 :r4 7
 :r5 8
:r4 ok
--
Odlišnosti v chování textových terminálů jsou
 :r1 sjednoceny instalací kompatibilního driveru
 :r2 transformovány na jednotné volání popisem v souboru /etc/termcap
 :r3 eliminovány společným voláním přes databázi v souboru /etc/termdbf
 :r4 omezeny na kompatibilní typy terminálů v compatibility seznamu
:r2 ok
--
Démon je
 :r1 je ovladač spících zařízení s nejnižší prioritou
 :r2 je spící program aktivovaný událostí
 :r3 infiltrovaný interpret příkazů
 :r4 ladící systém jádra operačního systému
:r2 ok
--
Každý řádek textového souboru v unixu končí
 :r1 znakem CR
 :r2 znakem LF
 :r3 znaky CR,LF
 :r4 znaky LF,CR
:r2 ok
--
Každý řádek textového souboru v unixu končí
 :r1 znakem Carriage Return
 :r2 znakem Long Field
 :r3 znaky CR,LF
 :r4 znaky LF,CR
 :r5 žádná z ostatních odpovědí není správná
:r5 ok
--
Každý řádek textového souboru v MS-DOSu/Windows končí
 :r1 znakem CR
 :r2 znakem LF
 :r3 znaky CR,LF
 :r4 znaky LF,CR
:r3 ok
--
Při přenosu textového souboru z MS-DOSu/Windows do unixu
 :r1 přidáváme řídící znak CR za každý řádek
 :r2 přidáváme řídící znak LF za každý řádek
 :r3 odebíráme řídící znak CR za každým řádkem
 :r4 odebíráme řídící znak LF za každým řádkem
:r3 ok
--
Při přenosu textového souboru z unixu do MS-DOSu/Windows
 :r1 přidáváme řídící znak CR za každý řádek
 :r2 přidáváme řídící znak LF za každý řádek
 :r3 odebíráme řídící znak CR za každým řádkem
 :r4 odebíráme řídící znak LF za každým řádkem
:r1 ok
--
Primární prompt běžného uživatele unixu je znak
 :r1 #
 :r2 $
 :r3 &
 :r4 *
 :r5 %
:r2 ok
--
Primární prompt superuživatele je znak

```
 :r1 #
 :r2 $
 :r3 &
 :r4 *
 :r5 %
:r1 ok
--
Primární prompt si uživatel nastavuje/mění v proměnné prostředí
 :r1 PROMPT
 :r2 PRMPT
 :r3 PRM1
 :r4 PS1
 :r5 PS
:r4 ok
--
Příkazem <TT>stty -a</TT>
 :r1 se uživatel odhlásí ze sezení u terminálu
 :r2 se uživateli obnoví výchozí nastavení terminálu
 :r3 se uživateli sdělí nastavení terminálu
 :r4 se uživateli zpřístupní všechny terminály
 :r5 se uživateli přepne terminál do ascii režimu
:r3 ok
--
Konvence pro prezentaci (výpis) znaku control-c je
 :r1 CTRL-C
 :r2 CC
 :r3 ^C
 :r4 C-C
 :r5 #C
:r3 ok
--
Výpis (prezentace) \012 znamená
 :r1 zápis hodnoty -12
 :r2 znak s ordinální hodnotou 12 dekadicky
 :r3 znak s ordinální hodnotou 12 oktalově
 :r4 znak s ordinální hodnotou 12 hexadecimálně
 :r5 textový řetězec 12
:r3 ok
--
Speciální znak <B>intr</B> (typicky control-c)
 :r1 přeruší činnost operačního systému
 :r2 přeruší běh procesu na popředí
 :r3 přeruší výpis na terminál na popředí
 :r4 přeruší výpis na terminál na pozadí
:r2 ok
--
Proces běžící na popředí násilně ukončím (např. pokud cyklí v nekonečné smyčce) stiskem speciálního znaku
 :r1 intr
 :r2 stop
 :r3 kill
 :r4 eof
:r1 ok
--
Speciální znak <B>eof</B> (typicky control-d)
 :r1 ukončí stav X-off
 :r2 přeruší běh procesu
 :r3 ukončí zpracování fail stavu
 :r4 vloží příznak konce souboru
:r4 ok
--
Speciální znaky <B>start</B> a <B>stop</B> (typicky control-q a control-s)
 :r1 realizují protokol X-ON/X-OFF
 :r2 spustí a ukončí proces
 :r3 přihlásí a odhlásí uživatele
 :r4 spustí a ukončí komunikaci operačního systému
:r1 ok
--
Při provádění příkazu <TT>cp /dev/tty /dev/null</TT> nezískáte prompt stisknutím speciálního znaku
 :r1 stop
 :r2 eof
 :r3 intr
 :r4 susp
:r1 ok
--
Co provede příkaz <TT>cp /etc/passwd /dev/tty</TT>
 :r1 zkopíruje obsah běžného souboru /etc/passwd do běžného souboru  /dev/tty
 :r2 zkopíruje obsah běžného souboru /etc/passwd na první tiskárnu systému
 :r3 zkopíruje obsah běžného souboru /etc/passwd na terminál uživatele
 :r4 čte z klávesnice terminálu a zapisuje do souboru /etc/passwd
:r3 ok
--
Přihlašovací (login) shell nelze za žádných okolností ukončit
 :r1 speciálním znakem eof
 :r2 speciálním znakem stop
 :r3 speciálním znakem intr
 :r4 příkazem exit
 :r5 rozpadnutím spojení
:r2 ok
--
Která služba používá šifrovanou komunikaci mezi klientem a serverem?
 :r1 telnet
 :r2 rsh
 :r3 ssh
 :r4 ftp
:r3 ok
--
Jaký počet kořenových adresářů je uživateli dostupný?
 :r1 tolik, kolik je v systému připojených diskových zařízení
 :r2 tolik, kolik je v systému připojených diskových zařízení, plus 1
 :r3 právě jeden
 :r4 žádný
:r3 ok
--
Při úspěšném přihlášení uživatele do systému se neprovede
 :r1 vypíší se systémové zprávy, jsou-li nějaké
 :r2 vypíše se sdělení správce systému, je-li nějaké
 :r3 spustí se uživatelem vybraný shell dle obsahu /etc/passwd
 :r4 uživatel se přihlásí do skupiny dle obsahu /etc/group
 :r5 vypíše se prompt shellu
:r4 ok
--
Při zadávání příkazů shellu neplatí, že
 :r1 lze příkazy zadávat "do zásoby"
 :r2 lze u některých shellů využít paměti starých (historii) příkazů
 :r3 příkazy lze zadávat bez ohledu na velikost písmen
 :r4 po stisknutí klávesy Enter již nelze opravovat obsah odeslaného řádku
 :r5 řádek lze upravovat tak dlouho, dokud není stisknuta klávesa Enter
:r3 ok
--
Při zadávání příkazů shellu klávesou Enter ještě neodešleme příkaz shellu ke zpracování (použijeme pokračovací řádek), když
 :r1 klávesu Enter stiskneme na konci řádku
 :r2 klávesu Enter stiskneme rychle dvakrát po sobě (tzv. doubleclick)
 :r3 před stiskem klávesy Enter vložíme obrácené lomítko
 :r4 po stisku klávesy Enter vložíme znak větší-než
:r3 ok
--
Sekundární prompt se používá
 :r1 ve vnořeném shellu
 :r2 na pokračovacím řádku
 :r3 při čtení příkazů ze skriptu
 :r4 na řádku zrušeném speciálním znakem kill
:r2 ok
--
Sekundární prompt lze přenastavit změnou proměnné prostředí PS2
a implicitně obsahuje znak
 :r1 $
 :r2 >
 :r3 #
 :r4 &
 :r5 *
:r2 ok
--
Historii mnou zadaných příkazů shellu bash mi vypíše příkaz
 :r1 man history
 :r2 cat history
 :r3 cat ~/.history
 :r4 cat ~/.bash_history
 :r5 žádná z ostatních odpovědí není správně
:r4 ok
--
Historie zadaných příkazů shellu bash se ukládá do souboru s historií příkazů, když
 :r1 se uživatel přihlašuje
 :r2 uživatel zadává jakýkoli příkaz
 :r3 se uživatel odhlašuje
 :r4 uživatel zadá příkaz history
:r3 ok
--
Operační systém Linux
 :r1 vznikl předáním ochranné známky UNIX společnosti GNU
 :r2 je předchůdcem unixu
 :r3 vznikl nezávisle na unixu
 :r4 je následovníkem unixu
:r3 ok
--
Uživatel ke v/v zařízením v unixu přistupuje prostřednictvím
 :r1 instalovaných driverů
 :r2 ikon zařízení
 :r3 speciálních souborů
 :r4 přerušení
:r3 ok
--
UNIX byl při svém vzniku koncipován jako systém
 :r1 multiprogramový, multiuživatelský, síťový
 :r2 multiprogramový, multiuživatelský, s terminálovým přístupem
 :r3 multiprogramový, síťový, s terminálovým přístupem
 :r4 multiuživatelský, síťový, s terminálovým přístupem
:r2 ok
--
Ukončí-li se shell, který byl spuštěn po přihlášení uživatele
 :r1 je uživatel odhlášen ze systému
 :r2 automaticky se spustí další shell, není-li zadán příkaz exit
 :r3 tento shell uživatel nemůže ukončit, protože je systémový
 :r4 systém se uživatele zeptá, zda chce spustit novou kopii shellu
:r1 ok
--
Bezprostředně po ukončení právě prováděného příkazu shellu, jsou-li zadány nějaké příkazy "do zásoby"
 :r1 se příkazy v zásobě ihned provedou bez potvrzení uživatele
 :r2 musí uživatel potvrdit provedení každého příkazu v zásobě zvlášť
 :r3 musí uživatel potvrdit celou skupinu příkazů v zásobě, jsou provedeny bud všechny nebo ani jeden
 :r4 nelze zadávat příkazy do zásoby bez povolení superuživatele
:r1 ok
--
Pro šifrované zpřístupnění terminálu vzdáleného počítače použijeme
 :r1 telnet
 :r2 ssh
 :r3 rsh
 :r4 ftp
 :r5 scp
:r2 ok
--
Které tři z vyjmenovaných aplikací patří mezi unixové shelly?
 :r1 bash, ssh, ksh
 :r2 ssh, telnet, putty
 :r3 sh, bash, ksh
 :r4 bash, sh-bash, ksh-bash
:r3 ok
--
Proces běžící na popředí pozastavím speciálním znakem
 :r1 intr
 :r2 eof
 :r3 susp
 :r4 supr
 :r5 su
```

:r3 ok
--
Nápovědu k příkazu <tt>rm</tt> získám pomocí
 :r1 rm -f
 :r2 rmman
 :r3 rm help
 :r4 man rm
:r4 ok
--
Proces je v unixu jednoznačně identifikován
 :r1 uživatelem, který spustil
 :r2 číslem přiřazeným při spuštění
 :r3 terminálem, ze kterého byl spuštěn
 :r4 časem spuštění
:r2 ok
--
Speciální znak <B>susp</B>
 :r1 přepne počítač do úsporného režimu
 :r2 pozastaví proces
 :r3 ukončí proces
 :r4 zamkne relaci uživatele
:r2 ok
--
Příkaz <B>ps</B>
 :r1 vypíše přihlášené uživatele
 :r2 vypíše spuštěné procesy
 :r3 vypíše programy v systému souborů
:r2 ok
--
Příkaz <tt>kill -9 1000</tt>
 :r1 násilně ukončí program s názvem 1000, mám-li na to právo
 :r2 násilně ukončí program s názvem 1000
 :r3 násilně ukončí proces s číslem 1000, mám-li na to právo
 :r4 násilně ukončí proces s číslem 1000
 :r5 je chybný příkaz
:r3 ok
--
Uživatel <B>root</B>
 :r1 smí číst všechny soubory, nesmí je však modifikovat
 :r2 smí číst všechny soubory a modifikovat jen ty, které jsou ve skupině <TT>root</TT>
 :r3 uživateli root se přístupová práva nekontrolují
:r3 ok
--
Uživatelů <B>root</B> smí být v systému zavedeno
 :r1 žádný
 :r2 jeden
 :r3 dva
 :r4 tolik, kolik jich je zavedeno v souboru /etc/passwd
:r2 ok
--
Příkaz <TT>man cat</TT>
 :r1 vypíše obsah souboru cat na standardní výstup
 :r2 vypíše obsah souboru man na standardní výstup
 :r3 vypíše manuálovou stránku příkazu cat na standardní výstup
 :r4 vypíše manuálovou stránku příkazu man na standardní výstup
 :r5 je chybný příkaz
:r3 ok
--
Příkaz <TT>man 5 passwd</TT>
 :r1 může vypsat manuálovou stránku příkazu passwd
 :r2 může vypsat manuálovou stránku souboru /etc/passwd
 :r3 může vypsat manuálovou stránku příkazu login
 :r4 je chybný příkaz
:r2 ok
--
Kterým příkazem nelze získat obsah (předpokládejme textového) souboru
 :r1 cat
 :r2 more
 :r3 less
 :r4 pwd
 :r5 všemi uvedenými
:r4 ok
--
Adresář /etc je určen
 :r1 pro dočasné soubory jako pomocný
 :r2 pro různé soubory, které nebylo možné dát do jiných adresářů
 :r3 pro umístění zpravidla konfiguračních souborů
 :r4 pro umístění zpravidla speciálních souborů
:r3 ok
--
Externí příkazy shellu jsou zpravidla umístěny v adresáři
 :r1 /usr/local/bin
 :r2 /bin
 :r3 /etc
 :r4 /usr
 :r5 /usr/lib
:r2 ok
--
Pro umísťování adresářů se soubory, jejichž obsah se často mění (např. log soubory se záznamy o provedených operacích), je určen adresář
 :r1 /bin
 :r2 /lib
 :r3 /usr
 :r4 /var
 :r5 /etc
:r4 ok

Za základní systém souborů považujeme, ten
 :r1 který je nejrozšířenější z podporovaných
 :r2 který je na hlavním disku systému
 :r3 který je na disku C:
 :r4 který obsahuje kořenový adresář systému
:r4 ok
--
Mezi základní systémy souborů patří
 :r1 iso8859-2, sysfs
 :r2 nfs, iso9660
 :r3 ext3, xfs
 :r4 vfat, tmpfs
:r3 ok
--
Mezi typické doplňující systémy souborů patří
 :r1 vfat, iso9660, ntfs
 :r2 swap, ext3, tmpfs
 :r3 ntfs, iso8859, xfs
:r1 ok
--
V souboru <TT>/etc/fstab</TT> je
 :r1 výstup kontroly systému souborů příkazem fsck
 :r2 seznam vadných bloků na zařízeních
 :r3 připojení systémů souborů k zařízením
 :r4 seznam uživateli dostupných systémů souborů
 :r5 takový soubor neexistuje
:r3 ok
--
Při neřízeném vypnutí stroje (např. při výpadku napájení) může dojít ke ztrátě dat zapisovaných do systému souborů
 :r1 plánovaných zapsat po výpadku
 :r2 během výpadku a plánovaných zapsat po výpadku
 :r3 před výpadkem, během výpadku a plánovaných zapsat po výpadku
:r3 ok
--
Operace spojená s poznačováním obsahu vyrovnávacích pamětí se nazývá typicky
 :r1 set
 :r2 save
 :r3 flush
 :r4 write
:r3 ok
--
Jméno souboru či adresáře smí být dlouhé nejvýše
 :r1 8+3 znaků
 :r2 16+3 znaků
 :r3 128 znaků
 :r4 255 znaků
:r4 ok
--
Maximální délka souboru v unixu je
 :r1 255 znaků
 :r2 256 znaků
 :r3 128 znaků
 :r4 je omezena vlastnostmi konkrétního systému souborů
:r4 ok
--
Soubory jejichž jméno začíná tečkou
 :r1 jsou přístupné pouze superuživateli
 :r2 se nezahrnují do *-expanze
 :r3 se nepoužívají
 :r4 takové jméno souboru není možné
:r2 ok
--
Příkaz <TT>ls -la</TT> nevypíše
 :r1 čas změny souboru
 :r2 číslo i-uzlu, ve kterém je uložen příslušný soubor
 :r3 velikost souboru
 :r4 soubory, jejichž jméno začíná tečkou
 :r5 adresářové položky . a ..
:r2 ok
--
Který příkaz vypíše počet odkazů na jednotlivé soubory?
 :r1 ls
 :r2 ls -l
 :r3 ls -a
 :r4 ls -i
:r2 ok
--
Relativní cesta k souboru nebo k adresáři začíná vždy
 :r1 lomítkem
 :r2 běžným adresářem
 :r3 domovským adresářem
 :r4 znakem ~
:r2 ok
--
Rozdíl mezi absolutní a relativní cestou k souboru či adresáři je
 :r1 absolutní cesta začíná vždy domovským adresářem
 :r2 absolutní cesta začíná vždy běžným adresářem
 :r3 absolutní cesta začíná vždy lomítkem
:r3 ok
--
K oddělování adresářů a jména souboru v zápisu cesty se používá znak
 :r1 \
 :r2 /
 :r3 |
:r2 ok
--
V hierarchii adresářů stojé nejvýše adresář
 :r1 /
 :r2 C:/
 :r3 C:/, D:/, E:/, ...
 :r4 root
 :r5 /root
:r1 ok
--
Běžný adresář jako domovský si uživatel nastaví
 :r1 příkazem cd
 :r2 editací souboru /etc/passwd
 :r3 příkazem pwd
 :r4 nenastaví
:r4 ok
--
Co se stane po provedení příkazu "<TT>cd -</TT>"?
Bude nastaven
 :r1 předchozí pracovní adresář
 :r2 domovský adresář aktuálního uživatele
 :r3 domovský adresář superuživatele
 :r4 speciální adresář definovaný v /etc/passwd
:r1 ok
--
Prázdný adresář je adresář
 :r1 je adresář s nulovou velikostí
 :r2 je adresář s prazdnou tabulkou adresáře
 :r3 obsahující dvě konkrétní položky
:r3 ok
--
Cestu k běžnému adresáři předá na standardní výstup příkaz
 :r1 cd
 :r2 pwd
 :r3 pcd
 :r4 wd
 :r5 grep $LOGNAME /etc/passwd|cut -d: -f6
:r2 ok
--
Jestliže v adresáři /home/user zadate příkaz <TT>cd ./..</TT> , potom váš běžný adresář bude

```
 :r1 /home
 :r2 /
 :r3 nelze zadat více argumentů, vypíše se
chyba
 :r4 /home/user
 :r5 domovský adresář
:r1 ok
--
Po zadaní příkazu <TT>cd /tmp/data</TT> v
adresáři /home/user se změní běžný adresář na
 :r1 /home/user/tmp/data
 :r2 /home/tmp/data
 :r3 /tmp/data
 :r4 /tmp/data/home/user
:r3 ok
--
Pracovní adresář je nastaven na /
home/user/data. Jaký bude pracovní adresář po
provedení příkazu <TT>cd ./../../.</TT>
 :r1 /
 :r2 /home
 :r3 /home/user
 :r4 /home/user/data
:r2 ok
--
Vytvoření a zrušení adresáře provedou příkazy
 :r1 md a rd
 :r2 mdir a rdir
 :r3 mkdir a rmdir
 :r4 make a remove
:r3 ok
--
Nemám-li povoleno v proměnné PATH
prohledávání běžného adresáře, spustím
proveditelný soubor <tt>muj</tt> příkazem
 :r1 run muj
 :r2 ../muj
 :r3 ./muj
 :r4 muj
:r3 ok
--
Adresář v systému souborů je uložen v souboru
 :r1 adresář není uložen v souboru
 :r2 obsahujícím čísla i-uzlu
 :r3 obsahujícím jména souborů a čísla i-uzlu
 :r4 obsahujícím cestu, jména souborů a čísla
i-uzlu
 :r5 obsahujícím identifikaci majitele,
cestu, jména souborů a čísla i-uzlu
:r3 ok
--
Kořenový adresář má číslo i-uzlu
 :r1 0
 :r2 1
 :r3 2
 :r4 3
 :r5 -1
:r3 ok
--
Kořenový adresář je v UNIXu označen
 :r1 /root
 :r2 \root
 :r3 \
 :r4 /
 :r5 C:\
:r4 ok
--
Položka '.' má v adresáři přiřazené číslo i-
uzlu
 :r1 0
 :r2 1
 :r3 2
 :r4 shodné s číslem i-uzlu tohoto adresář
 :r5 shodné s číslem i-uzlu rodičovského
adresáře
:r4 ok
--
Pokud je ve výpisu obsahu adresáře u adresáře
"." uvedeno číslo  i-uzlu 2 a u adresáře ".."
0, potom se jedná o
 :r1 běžný (pracovní) adresář
 :r2 nesmysl
 :r3 adresář druhé úrovně např. /home
 :r4 adresář třetí úrovně např. /home/user
:r2 ok
--
Příkazem rmdir se typicky maže adresář
obsahující
 :r1 0 položek
 :r2 1 položku
 :r3 2 položky
 :r4 3 položky
:r3 ok
--
i-uzel
 :r1 obsahuje adresu počítače v síti
 :r2 obsahuje informace o směrování paketů v
uzlech sítě
 :r3 obsahuje informaci o jednom souboru či
adresáři systému souborů
 :r4 obsahuje informaci o jednom uzlu n-
uzlového clusteru
:r3 ok
--
Zdrojový soubor zruší příkaz
 :r1 mv
 :r2 cp
 :r3 cat
 :r4 ls
:r1 ok
--
Soubory rušíme příkazem
 :r1 mv
 :r2 cp
 :r3 rm
 :r4 cat
 :r5 del
:r3 ok
--
Systém vždy fyzicky odstraní soubor z tabulky
i-uzlů (tzn. soubor se zruší), když
 :r1 provedu příkaz <tt>rm</tt>
 :r2 počet odkazů na počítadle v i-uzlu
klesne na 0
 :r3 se uživatel odhlásí, poté co provedl
příkaz <tt>rm</tt>
 :r4 se vysype koš
 :r5 fyzicky zruším i-uzel příkazem rmnode
:r1 příkazem <tt>rm</tt> odstraníte jen jeden
z odkazů, další ještě mohou existovat
:r2 ok
--
Co se stane po provedení příkazu "<TT>rm -rf
*</TT>"?
Bude smazán obsah pracovního adresáře
 :r1 vč. jmen začínajících tečkou a vč.
podadresářů a bez všech varování
 :r2 vč. jmen začínajících tečkou, ale bez
podadresářů
 :r3 bez jmen začínajících tečkou, vč.
podadresářů a bez všech varování
 :r4 bez jmen začínajících tečkou, bez
podadresářů a bez všech varování
 :r5 vč. jmen začínajících tečkou a vč.
podadresářů s varováními
:r3 ok
--
i-uzel obsahuje
 :r1 všechny informace o souboru vyjma dat
souboru
 :r2 všechny informace o souboru vyjma cesty
k souboru a dat
 :r3 všechny informace o souboru vyjma jména
souboru, cesty k souboru a dat
 :r4 všechny informace o souboru vyjma
identifikace majitele, jména souboru, cesty k
souboru a dat
 :r5 všechny informace o tomto uzlu v síti
:r3 ok
--
Kořenový adresář má pro položky '.' a '..'
 :r1 čísla i-uzlu o jedničku větší
 :r2 čísla i-uzlu o jedničku menší
 :r3 čísla i-uzlu stejná
 :r4 žádná z ostatních odpovědí není správná
:r3 ok
--
Pokud příkazem <TT>mv</TT> přesouváte soubory
v rámci jednoho systému souborů, pak se číslo
i-uzlu přesouvaného souboru
 :r1 změní
 :r2 nezmění
 :r3 zruší
:r2 ok
--
Co provede příkaz <TT>ls -a</TT> (označte
nejsprávnější odpověď):
 :r1 vypíše seznam všech podadresářů
 :r2 vypíše seznam všech souborů
 :r3 vypíše seznam všech podadresářů a
souborů a položek se jménem začínajícím
tečkou
 :r4 vypíše seznam všech položek se jménem
začínajícím tečkou
 :r5 provede totéž jako příkaz ls bez
parametru
:r3 ok
--
Kterým příkazem lze smazat neprázdný adresář?
 :r1 rmdir
 :r2 rm -r
 :r3 rm -a
 :r4 rm -f
 :r5 neprázdný adresář nelze smazat
:r2 ok
--
Který příkaz předá na standardní výstup obsah
proměnné PATH?
 :r1 show $PATH
 :r2 echo $PATH
 :r3 echo PATH
 :r4 ls $PATH
:r2 ok
--
Příkazem <TT>pwd</TT>
 :r1 změníme pracovní adresář
 :r2 vypíšeme pracovní adresář
 :r3 změníme domovský adresář
 :r4 vypíšeme domovský adresář
:r2 ok
--
Příkazem <TT>rmdir</TT>
 :r1 vymažeme prázdný adresář
 :r2 přesuneme prázdný nebo neprázdný adresář
do koše
 :r3 přesuneme prázdný adresář do koše
 :r4 vyprázdníme koš
 :r5 vymažeme skrytý adresář
:r1 ok
--
Maximální počet i-uzlů v systému souborů
 :r1 lze dle potřeby měnit příkazem ls -il
 :r2 lze dle potřeby měnit příkazem mkfs
 :r3 nelze měnit bez nového vytvoření systému
souborů
:r3 ok
--
Počet odkazů na soubor je uložen
 :r1 v adresáři
 :r2 v i-uzlu
 :r3 v souboru
 :r4 není uložen, vždy se při potřebě
vypočítává
:r2 ok
--
Na soubor vytvořený běžným způsobem textovým
editorem
 :r1 je nulový počet tvrdých odkazů
 :r2 je jeden tvrdý odkaz
 :r3 jsou dva tvrdé odkazy
:r2 ok
--
Jakým příkazem vytvoříme tvrdý odkaz na
soubor?
 :r1 ln -t soubor odkaz
 :r2 ln -h soubor odkaz
 :r3 ln soubor odkaz
 :r4 ln -s soubor odkaz
:r3 ok
--
Po provedení posloupnosti příkazů
<pre>touch a; ln a b; ln a c</pre>
vzniknou následující počty tvrdých odkazů na
jednotlivé soubory
 :r1 a 1, b 2, c 3
 :r2 a 3, b 2, c 1
 :r3 a 2, b 2, c 2
 :r4 a 3, b 3, c 3
 :r5 příkaz je chybný
:r4 ok
--
Kolik tvrdých odkazů ukazuje na soubor v
následujícím výpisu příkazu <TT>ls -l</TT>?
<PRE>-rw-r--r--  3 ja oni  2 Mar 1 04:05
cosi</PRE>
 :r1 1
 :r2 2
 :r3 3
 :r4 4
 :r5 5
:r3 ok
--
Po provedení příkazu <TT>ln aa bb</TT>
přibude v běžném adresáři na výpise příkazu
<TT>ls -l</TT> položka
 :r1 lrw-r--r--  1 brandejs staff  6 dub  8
```

21:50 bb
 :r2 -rw-r--r-- 1 brandejs staff 6 dub 8
21:50 bb
 :r3 lrw-r--r-- 2 brandejs staff 6 dub 8
21:50 bb
 :r4 -rw-r--r-- 2 brandejs staff 6 dub 8
21:50 bb
:r4 ok
--
Výpis příkazu <TT>ls -il</TT>:
<PRE>539520038 -rw-r--r-- 2 brandejs staff 0 bře 29 13:46 a
539520038 -rw-r--r-- 2 brandejs staff 0 bře 29 13:46 b</PRE>
Který z odkazů vznikl dříve?
 :r1 odkaz na soubor a
 :r2 odkaz na soubor b
 :r3 vznikly zároveň
 :r4 nelze určit
 :r5 výpis je nesmyslný
:r4 ok
--
Výpis příkazu <TT>ls -il</TT>:
<PRE>539520038 -rw-r--r-- 2 brandejs staff 0 bře 29 13:46 a
539520038 -rw-r--r-- 2 brandejs staff 0 bře 29 13:46 b</PRE>
Kdo vytvořil tvrdý odkaz 'a' na 'b' nebo 'b' na 'a'?
 :r1 výhradně uživatel brandejs
 :r2 výhradně uživatel staff
 :r3 výhradně člen skupiny staff
 :r4 výhradně člen skupiny brandejs
 :r5 nelze určit
:r5 ok
--
Který příkaz úspěšně skončí po provedení posloupnosti příkazů
<PRE>touch a; mkdir b</PRE>
 :r1 ln a c
 :r2 ln b c
 :r3 žádný z uvedených příkazů
:r1 ok
--
Uživatel xnovak vytvořil příkazem <B>ln</B> tvrdý odkaz na soubor, který vlastní uživatel xpolak. Kdo bude ve výpise příkazu <TT>ls -l</TT> uveden jako vlastník odkazu?
 :r1 xnovak
 :r2 xpolak
 :r3 oba
 :r4 nikdo z nich
:r2 ok
--
Kolik bude tvrdých odkazů na adresář a, provedete-li posloupnost příkazů
<PRE>mkdir a; mkdir a/b; mkdir a/c; touch a/d</PRE>
 :r1 2
 :r2 3
 :r3 4
 :r4 5
 :r5 posloupnost příkazů je nesmyslná
:r3 ok
--
Kolik bude tvrdých odkazů na adresář a, provedete-li posloupnost příkazů
<PRE>touch a; touch a/b; touch a/c; touch a/d</PRE>
 :r1 2
 :r2 3
 :r3 4
 :r4 5
 :r5 posloupnost příkazů je nesmyslná
:r5 ok
--
Kolik bude tvrdých odkazů na adresář b, provedete-li posloupnost příkazů
<PRE>mkdir a; mkdir a/b; mkdir a/c; mkdir a/b/d; mkdir a/c/e</PRE>
 :r1 2
 :r2 3
 :r3 4
 :r4 5
 :r5 posloupnost příkazů je nesmyslná
:r2 ok
--
Kolik bude tvrdých odkazů na adresář b, provedete-li posloupnost příkazů
<PRE>mkdir a; mkdir a/b; mkdir a/c; mkdir a/b/d; rmdir a/b/d</PRE>
 :r1 2
 :r2 3
 :r3 4
 :r4 5
 :r5 posloupnost příkazů je nesmyslná
:r1 ok
--
Po provedení posloupnosti příkazů
<PRE>mkdir a; mkdir a/b; ls a; ls -d a</PRE>
se na standardní výstup předají řádky obsahující
 :r1 a
b
 :r2 b
a
 :r3 a
a
 :r4 b
b
 :r5 posloupnost příkazů je nesmyslná
:r2 ok
--
Po provedení posloupnosti příkazů
<PRE>touch a; mkdir b; touch b/a; ls a; ls b</PRE>
se na standardní výstup předají řádky obsahující
 :r1 a
b
 :r2 b
a
 :r3 a
a
 :r4 b
b
 :r5 posloupnost příkazů je nesmyslná
:r3 ok
--
Z výpisu příkazu <TT>ls -l</TT>
<PRE> -rwxr-xr-x 2 brandejs staff 6 bře 29 23:05 a
 drwxr-xr-x 2 brandejs staff 6 bře 29 23:05 b
 -rwxr-xr-x 2 brandejs staff 6 bře 29 23:05 c</PRE>
vyplývá, že
 :r1 a, c jsou soubory, b je speciální soubor
 :r2 a, c jsou soubory, b je adresář
 :r3 a, b jsou adresáře, c je soubor
 :r4 a, c jsou adresáře, b je soubor
 :r5 výpis je nesmyslný
:r2 ok
--
Tvrdý odkaz se nepovolí udělat na
 :r1 speciální soubor
 :r2 symbolický odkaz
 :r3 adresář
 :r4 soubor
:r3 ok
--
Symbolický odkaz se nepovolí udělat na
 :r1 adresář
 :r2 soubor v jiném systému souborů
 :r3 kořenový adresář
 :r4 speciální soubor /dev/null
 :r5 i-uzel
:r5 ok
--
Provedením příkazu ln (bez voleb) se počet odkazů na objekt typicky
 :r1 zvyšuje
 :r2 snižuje
 :r3 nemění
:r1 ok
--
Provedením příkazu rm se počet odkazů na objekt typicky
 :r1 zvyšuje
 :r2 snižuje
 :r3 nemění
:r2 ok
--
Provedením příkazu ln -s se počet odkazů na objekt typicky
 :r1 zvyšuje
 :r2 snižuje
 :r3 nemění
:r3 ok
--
Příznak, že objekt je symbolický odkaz, je uveden
 :r1 v adresáři
 :r2 v i-uzlu
 :r3 v souboru
 :r4 není uveden nikde
:r2 ok
--
Z výpisu příkazu <TT>ls -l</TT>
<PRE>-rw-r--r-- 2 brandejs staff 0 bře 29 23:35 b
lrwxrwxrwx 1 brandejs staff 1 bře 29 23:35 c -> a</PRE>
vyplývá, že
 :r1 a je symbolický odkaz, b je tvrdý odkaz
 :r2 c je symbolický odkaz, b je tvrdý odkaz
 :r3 b je symbolický odkaz, a je tvrdý odkaz
 :r4 b je symbolický odkaz, c je tvrdý odkaz
 :r5 výpis je nesmyslný
:r2 ok
--
Jaká je velikost dat souboru se symbolickým odkazem ukazujícím na soubor <TT>mujprog.c</TT>
 :r1 žádná data se k symbolickému odkazu neukládají
 :r2 velikost dat je shodná s velikostí souboru <TT>mujprog.c</TT>
 :r3 0 bajtů
 :r4 9 bajtů
:r4 ok
--
Co se předá na standardní výstup po provedení posloupnosti příkazů
<PRE>touch a;ln -s a b;rm a;ls</PRE>
 :r1 a
 :r2 b
 :r3 posloupnost příkazů je chybná, protože nelze smazat soubor, na který ukazuje symbolický odkaz
:r2 ok
--
Dynamickou informaci o uzamknutí souboru (výlučný přístup) obsahuje
 :r1 i-uzel na disku
 :r2 paměťová kopie i-uzlu
 :r3 je uložena mimo i-uzly
:r2 ok
--
Informaci o přístupových právech souboru obsahuje
 :r1 i-uzel na disku
 :r2 výhradně paměťová kopie i-uzlu
 :r3 je uložena mimo i-uzly
:r1 ok
--
Superblok je
 :r1 oblast pro uložení dat souborů
 :r2 informační struktura o systému souborů
 :r3 náhradní blok za vadné sektory na disku
 :r4 blok dat se zašifrovanou informací
:r2 ok
--
Speciální soubor je
 :r1 totéž co soubor 'skrytý'
 :r2 soubor přístupný pouze superuživateli
 :r3 zpřístupnění v/v zařízení
 :r4 zvláštní soubor jako adresář, symbolický odkaz apod.
 :r5 soubor se speciálními právy
:r3 ok
--
Speciální soubory jsou ve výpisu příkazu <TT>ls -l</TT> identifikovány v prvním sloupci znakem nebo znaky
 :r1 k, l
 :r2 s
 :r3 x
 :r4 b, c
 :r5 -
:r4 ok
--
Typickým příkladem speciálního souboru je
 :r1 /
 :r2 /etc/passwd
 :r3 /var/mail/xnovak
 :r4 /dev/null
 :r5 /bin/ls
:r4 ok
--
Blokový nebo znakový typicky je
 :r1 adresář
 :r2 přístupový kód
 :r3 systém souborů
 :r4 speciální soubor
 :r5 superblok
:r4 ok
--
Pojmenovaná roura
 :r1 je způsob přenášení dat mezi dvěma počítači
 :r2 je speciální soubor pro přenos dat mezi dvěma procesy
 :r3 je prázdné zařízení, tzv. koš na bity

:r4 připojovací bod pro zpřístupnění připojeného systému souborů
:r2 ok
--
Která posloupnost představuje korektní vytvoření a použití pojmenované roury?
:r1 mkdir roura p;cat /dev/null > roura & cat roura; rm roura
:r2 mknod roura p;cat /etc/passwd > roura & cat roura; rm roura
:r3 mknod roura p;cat /etc/passwd > roura & cat roura; rmnod roura
:r4 mknod roura p; cat roura > /etc/passwd & cat /etc/passwd; rmnod roura
:r5 cat roura > /etc/passwd & mknod roura p; cat /etc/passwd; rm roura
:r2 ok
--
Speciální soubory pro zpřístupnění V/V zařízení se typicky ukládají do adresáře
:r1 /specf
:r2 /iofiles
:r3 /io
:r4 /dev
:r5 /devio
:r4 ok
--
Který příkaz přenese nejvíce dat?
:r1 cp /etc/passwd /tmp/mypasswd; cat /dev/null > /tmp/mypasswd
:r2 cp /etc/passwd /tmp/mypasswd; cat /tmp/mypasswd > /dev/null
:r3 cp /etc/passwd /dev/null; cat /dev/null > /dev/null
:r4 cp /etc/passwd /dev/null; cat /dev/null > /dev/null
:r2 ok
--
Jakou bude mít velikost soubor 'x' po provedení příkazů
<TT>echo 'ahoj' > x; cp /dev/null x</TT>
:r1 nebude existovat
:r2 0
:r3 1
:r4 4
:r5 5
:r2 ok
--
Přístupová práva k objektům systému souborů se určují zvlášť pro
:r1 vlastníka, členy skupiny, ostatní přihlášené uživatele
:r2 uživatele, členy skupiny, ostatní nepřihlášené uživatele
:r3 uživatele, členy skupiny, ostatní přihlášené uživatele
:r4 vlastníka, členy skupiny, ostatní nepřihlášené uživatele
:r1 ok
--
Přístupová práva se nastavují v pořadí a s počátečními písmeny zkratky
:r1 owner, group, world
:r2 root, world, execute
:r3 user, root, others
:r4 user, group, others
:r4 ok
--
Přístupová práva k souboru se určují zvlášť pro trojici operací
:r1 čtení, zrušení, spuštění
:r2 provedení, zrušení, kopírování
:r3 zrušení, spuštění, editace
:r4 čtení, zápis, provedení
:r5 modifikace, zkrácení, spuštění
:r4 ok
--
Přístupová práva k adresáři se definují zvlášť pro trojici operací
:r1 vytváření, rušení, spouštění
:r2 vstup do adresáře, zrušení, vytvoření
:r3 čtení, zapisování, vstup do adresáře
:r4 čtení, vytváření, rušení
:r5 čtení, spouštění, vstup do adresáře
:r3 ok
--
Co znamená přístupové právo "x" pro soubor?
:r1 ze souboru lze číst
:r2 do souboru lze zapisovat
:r3 soubor lze spustit
:r4 do souboru lze vstoupit
:r3 ok
--
Co znamená přístupové právo "x" pro adresář?
:r1 adresář můžeme vypsat
:r2 do adresáře můžeme zapisovat
:r3 do adresáře je povoleno vstoupit
:r4 adresář je povoleno spustit
:r3 ok
--
Přístupová práva k souboru či adresáři jsou uložena
:r1 v jednom bajtu
:r2 ve dvou bajtech
:r3 na devíti bitech
:r4 ve dvanácti bitech
:r5 ve 128 bajtech
:r4 ok
--
Vytvořit nový soubor smím v adresáři, na který mám právo alespoň
:r1 čtení a zápisu
:r2 vytváření a vstupu
:r3 zápisu a vstupu
:r4 čtení a vytváření
:r5 čtení, zápisu a vstupu
:r3 ok
--
Zrušit soubor v adresáři smím, pokud mám alespoň
:r1 právo zápisu do souboru
:r2 právo zápisu do adresáře a zápisu do souboru
:r3 právo zápisu a vstupu do adresáře
:r4 právo zápisu a vstupu do adresáře, a právo zápisu do souboru
:r5 právo zápisu do souboru a jsem vlastníkem souboru
:r3 ok
--
Jaké právo k adresáři musím alespoň mít, pokud chci vytvořit soubor v tomto adresáři?
:r1 -wx
:r2 rw-
:r3 r-x
:r4 -w-
:r5 rwx
:r1 ok
--
V adresáři, který je na výpise příkazu <TT>ls -l</TT> označen <TT>drwxrwxrwx</TT>, smím
:r1 rušit libovolné soubory
:r2 rušit soubory, které vlastním
:r3 rušit soubory, ke kterým mám právo zápisu
:r4 nesmím rušit soubory
:r5 rušit soubory, které vlastním a ke kterým mám právo zápisu
:r1 ok
--
V adresáři, který je na výpise příkazu <TT>ls -l</TT> označen <TT>drwxrwxrwt</TT>, smím
:r1 rušit libovolné soubory
:r2 rušit soubory, které vlastním
:r3 rušit soubory, ke kterým mám právo zápisu
:r4 nesmím rušit soubory
:r5 rušit soubory, které vlastním a ke kterým mám právo zápisu
:r2 ok
--
V adresáři, který je na výpise příkazu <TT>ls -l</TT> označen <TT>d--x--x--x</TT>, smím
:r1 rušit soubor
:r2 vytvořit soubor
:r3 podadresář nastavit jako běžný
:r4 vypsat jeho obsah příkazem <TT>ls</TT>
:r5 žádná jiná odpověď není správná
:r3 ok
--
Soubor mohu číst, pokud mám právo alespoň
:r1 čtení souboru
:r2 vstupu do adresáře a čtení souboru
:r3 čtení adresáře, vstupu do adresáře a čtení souboru
:r4 čtení adresáře, vstupu do adresáře a čtení souboru a vstupu do souboru
:r2 ok
--
Do souboru mohu zapisovat, pokud mám právo alespoň
:r1 zapisovat do adresáře
:r2 zapisovat do souboru
:r3 vstupovat do adresáře a zapisovat do souboru
:r4 zapisovat a vstupovat do adresáře, zapisovat a číst soubor
:r3 ok
--
Soubor (s proveditelným binárním kódem) smím spustit, pokud mám právo alespoň
:r1 na čtení souboru
:r2 čtení adresáře a spuštění souboru
:r3 vstup do adresáře a čtení souboru
:r4 vstup do adresáře a spuštění souboru
:r5 vstup do adresáře, spuštění souboru a čtení souboru
:r4 ok
--
Proces, který spustím ze spustitelného souboru obsahujícího proveditelný binární kód a majícího nastavený SUID bit, poběží vždy pod identifikací (uživatelem)
:r1 uživatele, který vlastní soubor
:r2 uživatele, který proces spustil
:r3 superuživatele
:r1 ok
--
Proces, který spustím ze spustitelného souboru obsahujícího proveditelný binární kód a nemajícího nastavený SUID bit, poběží vždy pod identifikací (uživatelem)
:r1 uživatele, který vlastní soubor
:r2 uživatele, který proces spustil
:r3 superuživatele
:r2 ok
--
Který soubor lze spustit tak, že poběží pod identifikací vlastníka souboru (vypsána jsou přístupová práva z příkazu <TT>ls -l</TT>)
:r1 rwxr-Sr-x
:r2 rws--x--x
:r3 rwxr-sr-x
:r2 ok
--
Který soubor lze spustit tak, že poběží pod identifikací uživatele, který jej spustil (vypsána jsou přístupová práva z příkazu <TT>ls -l</TT>)
:r1 -wS--S---
:r2 rws--x--x
:r3 rwxr-xr-x
:r3 ok
--
Co znamená SUID bit?
:r1 příznak superuživatele v souboru /etc/passwd
:r2 příznak vlastníka adresáře systému souborů
:r3 příznak měnící identifikaci majitele procesu
:r3 ok
--
Jsem-li vlastníkem souboru - s jakými přístupovými právy smím soubor spustit (jedná se o spustitelnou binárku; přístupová práva jsou zapsána osmičkově)?
:r1 0644
:r2 0766
:r3 2264
:r4 7666
:r2 ok
--
Jsem-li členem skupiny souboru, ne jeho vlastníkem - s jakými přístupovými právy smím soubor spustit (jedná se o spustitelnou binárku; přístupová práva jsou zapsána osmičkově)?
:r1 0644
:r2 0766
:r3 2264
:r4 3656
:r4 ok
--
Přístupová práva zapsaná osmičkově 4522 znamenají
:r1 r-s-w--w-
:r2 -wS--x--x
:r3 rw-r--r-t
:r4 rwx-ws-w-
:r5 r--r-x-w-
:r1 ok
--
Přístupová práva zapsaná osmičkově 0731 znamenají
:r1 ---rwx-wx

```
 :r2 rwxrw-r--
 :r3 rwx-wx--x
 :r4 r-xrwx--r
 :r5 --r-wxrwx
:r3 ok
--
Symbol 't' v přístupových právech znamená
 :r1 objekty v adresáři smí rušit jen
vlastník objektu nebo vlastník adresáře
 :r2 objekty v adresáři smí rušit jen
vlastník objektu
 :r3 objekty v adresáři vytvářet jen vlastník
adresáře
 :r4 vlastníkem nově vytvořeného objektu bude
vlastník adresáře
:r1 ok
--
Tzv. sticky bitem zapínáme následující
chování
 :r1 soubory v adresáři smí rušit pouze
jejich vlastník
 :r2 soubory v adresáři smí rušit pouze
superuživatel
 :r3 soubory v adresáři smí rušit pouze
jejich vlastník a vlastník adresáře
 :r4 soubory v adresáři smí vytvářet pouze
vlastník adresáře
 :r5 soubory v adresáři smí vytvářet pouze
vlastník adresáře nebo superuživatel
:r3 ok
--
Jaká podmínka je postačující k tomu, abych
mohl měnit přístupová práva souboru?
 :r1 jsem vlastníkem souboru a mám právo
zápisu do souboru
 :r2 mám právo zápisu do souboru
 :r3 mám právo zápisu do souboru a zápisu do
adresáře
 :r4 jsem vlastníkem souboru
 :r5 mám právo zápisu do adresáře
:r4 ok
--
Jaká syntaxe příkazu chmod je chybná?
 :r1 chmod go+rx
 :r2 chmod go=o
 :r3 chmod go+X
 :r4 chmod u-s
 :r5 chmod x+a
:r5 ok
--
Kterým příkazem přidáme ke všem položkám
běžného adresáře právo zápisu pro skupinu
uživatelů? (Ostatní práva neměňte.)
 :r1 chmod g+w *
 :r2 chmod w+g *
 :r3 chmod w=g *
 :r4 chmod g=w *
 :r5 chmod w+gx *
:r1 ok
--
Kterým příkazem přidáme ke všem položkám
běžného adresáře právo čtení pro (úplně)
všechny uživatele? (Ostatní práva neměňte.)
 :r1 chmod r=a *
 :r2 chmod a+r *
 :r3 chmod go+w *
 :r4 chmod r+a *
 :r5 chmod ugo+x *
:r2 ok
--
Kterým příkazem přidáte všem spustitelným
souborům a všem adresářům právo v běžném
adresáři právo spuštění/vstupu pro všechny
ostatní? (Práva jiných souborů a jiná práva
neměňte.)
 :r1 chmod o=x *
 :r2 chmod o+x *
 :r3 chmod o+X *
 :r4 chmod g+x *
 :r5 nelze udělat
:r3 ok
--
Jakým příkazem nastavíme SUID bit?
 :r1 chmod u+s
 :r2 chmod g+s
 :r3 chmod o+s
 :r4 chmod x+s
 :r5 chmod s+g
:r1 ok
--
Jakým příkazem nastavíme přístupová práva ve
tvaru <TT>rw-r--r--</TT>
 :r1 chmod 640
 :r2 chmod ugo=640
 :r3 chmod a=rw,go-w
:r3 ok
--
Jaká práva bude mít soubor, pokud provedu
příkaz <TT>chmod 6755 soubor</TT>
 :r1 rwsr-sr-x
 :r2 rwsrwsr-x
 :r3 rwsrw-rw-
 :r4 rwxrwsr-x
:r1 ok
--
K čemu slouží příkaz <TT>chmod g=u</TT>
 :r1 nastaví stejné práva pro skupinu jako má
vlastník
 :r2 nastaví stejné práva pro vlastníka jako
má skupina
 :r3 zruší všechny práva u vlastníka a
skupiny
 :r4 žádná z uvedených odpovědí
:r1 ok
--
Jsem členem více než jedné skupiny. Kdy má
smysl, abych použil příkaz <B>newgrp</B>
 :r1 chci-li vytvořit novou skupinu v systému
 :r2 chci-li se dostat k souborům patřícím do
jiné skupiny, než do které se chci přihlásit
 :r3 chci-li při vytvoření nového souboru
použít jinou skupinu, než do které jsem
přihlášen
 :r4 chci-li požádat administrátora o mé
přihlášení do nové skupiny
:r3 ok
```

```
Řídícím operátorem není
 :r1 //
 :r2 ||
 :r3 &&
 :r4 ;;
:r1 ok
--
Znak, který následuje za \ se
 :r1 nepovažuje za řídící, a pokud za ním
následuje znak "nový řádek", pokračuje se na
dalším řádku.
 :r2 nepovažuje za řídící, a pokud za ním
následuje znak "nový řádek", nepokračuje se
na dalším řádku.
 :r3 považuje za řídící, a pokud za ním
následuje znak "nový řádek", pokračuje se na
dalším řádku.
 :r4 považuje za řídící, a pokud za ním
následuje znak "nový řádek", nepokračuje se
na dalším řádku.
:r1 ok
--
Znaky uzavřené do dvojice apostrofů ('...')
ztrácejí svůj řídící význam s výjimkou
 :r1 ' (apostrof)
 :r2 $, ` (obrácený apostrof), \ (následuje-
li $, `, ", \, nový řádek)
 :r3 ' (apostrof), \
 :r4 ` (obrácený apostrof)
:r1 ok
--
Znaky uzavřené do dvojice úvozovek ("...")
ztrácejí svůj řídící význam s výjimkou
 :r1 ' (apostrof)
 :r2 $, ` (obrácený apostrof), \ (následuje-
li $, `, ", \, nový řádek)
 :r3 ' (apostrof), \
 :r4 ` (obrácený apostrof)
:r2 ok
--
Adresář se jménem "Ferda mravenec" (má ve
jméně mezeru) nevytvořím příkazem
 :r1 mkdir "Ferda mravenec"
 :r2 mkdir 'Ferda mravenec'
 :r3 mkdir `Ferda mravenec`
 :r4 mkdir Ferda\ mravenec
 :r5 mkdir Ferda" "mravenec
:r3 ok
--
Co se předá na standardní výstup po spuštění
příkazu<PRE>echo \'</PRE>
 :r1 Bad filename
 :r2 echo \'
 :r3 \'
 :r4 ''
 :r5 '
:r5 ok
--
Co se předá na standardní výstup po spuštění
příkazu echo \'\\\"\'
 :r1 '\"'
 :r2 '\\\"'
 :r3 \\\"\'
 :r4 \\\"
:r1 ok
--
Příkaz při ukončení vrací ukončovací kód.
Jaký ukončovací kód je?
 :r1 číselný; 0 znamená úspěšné ukončení,
nenulové znamená ukončení s chybou
 :r2 číselný; 1 znamená úspěšné ukončení, 0
znamená ukončení s chybou
 :r3 číselný; nenulový znamená úspěšné
ukončení, 0 znamená ukončení s chybou
 :r4 textový; 'ok' znamená úspěšné ukončení,
prázdný řetězec znamená ukončení s chybou
 :r5 textový; prázdný řetězec znamená úspěšné
ukončení,  neprázdný řetězec znamená ukončení
s chybou
:r1 ok
--
Znak \ má následující význam:
 :r1 oddělovač adresářů a jména souboru od
adresáře v cestě
 :r2 zapnutí pokračovacího řádku
 :r3 označení control znaku
 :r4 zrušení řídicího charakteru
následkujícího znaku
:r2 pokračovací řádek zapíná kombinace \ a
nový řádek
:r4 ok
--
Znak ' (apostrof) má následující význam:
 :r1 všechny znaky v řetězci uzavřeném do
dvojice '...' ztrácí řídicí význam

 :r2 příkaz uzavřený do dvojice '...' se při
expanzi příkazového řádku provede a celý
řetězec se nahradí obsahem standardního
výstupu
 :r3 znaky v řetězci uzavřeném do dvojice
'...' ztrácí řídicí význam vyjma řídicích
znaků $`\
 :r4 označuje control znak
:r1 ok
--
Znak " má následující význam:
 :r1 všechny znaky v řetězci uzavřeném do
dvojice "..." ztrácí řídicí význam
 :r2 příkaz uzavřený do dvojice "..." se při
expanzi příkazového řádku provede a celý
řetězec se nahradí obsahem standardního
výstupu
 :r3 znaky v řetězci uzavřeném do dvojice
"..." ztrácí řídicí význam vyjma řídicích
znaků $`\
 :r4 označuje control znak
:r3 ok
--
Kolik obrácených lomítek a kolik apostrofů
předá na standardní výstup příkaz <TT>echo
'\'"'"\\\'</TT>
 :r1 2 a 1
 :r2 2 a 2
 :r3 3 a 2
 :r4 3 a 3
 :r5 4 a 4
:r2 ok
--
Který z příkazů nesmaže soubor x?
 :r1 rm x
 :r2 rm 'x'
 :r3 rm "x"
 :r4 rm `x`
 :r5 rm \x
:r4 ok
--
Příkaz<PRE>sort > soubor1 < soubor2</PRE>
 :r1 čte standardní vstup ze souboru
<TT>soubor2</TT> a standardní výstup zapisuje
do souboru <TT>soubor1</TT>
 :r2 čte standardní vstup ze souboru
<TT>soubor1</TT> a standardní výstup zapisuje
do souboru <TT>soubor2</TT>
 :r3 čte standardní vstup ze souboru
<TT>soubor2</TT> a standardní chybový výstup
zapisuje do souboru <TT>soubor1</TT>
```

:r4 čte standardní vstup ze souboru <TT>soubor1</TT> a standardní chybový výstup zapisuje do souboru <TT>soubor2</TT>
:r1 ok
--
Operátory přesměrování jsou
:r1 >&, >, >>, <<-, <>
:r2 <<, >&, <->, <&
:r3 <, >, >|, <=, >>>
:r4 >>, <<, <$, <, >, $>
:r1 ok
--
Který z příkazů nezkopíruje soubor 'a' do souboru 'b'?
:r1 cp a b
:r2 cat a > b
:r3 cat < a > b
:r4 cat a|cat > b
:r5 cp < a > b
:r5 ok
--
Jaké je korektní a smysluplné přesměrování vstupu?
:r1 ls < adresar
:r2 cd < adresar
:r3 grep Brno < mesta
:r4 ls > adresar
:r3 ok
--
Jaké je korektní a smysluplné přesměrování výstupu?
:r1 echo toto se mi libi > soubor
:r2 cd adresar > soubor
:r3 mv a > b
:r4 grep Brno < mesta
:r1 ok
--
Soubor 'a' před provedením příkazu <TT>ls > a</TT>
:r1 musí existovat
:r2 nesmí existovat
:r3 může existovat
:r3 ok
--
Soubor 'a' před provedením příkazu <TT>sort < a</TT>
:r1 musí existovat
:r2 nesmí existovat
:r3 může existovat
:r1 ok
--
Po provedení příkazu <PRE>echo je > je; rm -rf neni; ls je neni > a 2> b</PRE>
:r1 bude soubor 'a' větší než soubor 'b'
:r2 bude soubor 'b' větší než soubor 'a'
:r3 budou soubory 'a' a 'b' stejně velké
:r4 soubor 'b' nebude existovat nebo bude prázdný
:r5 soubor 'a' nebude existovat nebo bude prázdný
:r2 ok
--
Po provedení příkazu <PRE>echo a > b > c</PRE>
:r1 bude soubor 'b' prázdný a soubor 'c' neprázdný
:r2 budou soubory 'b' a 'c' stejně velké
:r3 bude soubor 'b' neprázdný a soubor 'c' prázdný
:r4 soubor 'b' nebude existovat
:r5 soubor 'a' se vyprázdní
:r1 ok
--
Po provedení příkazu <PRE>echo b > s; echo a >> s;sort < s > s</PRE>
:r1 bude soubor 's' prázdný
:r2 bude soubor 's' obsahovat řádky v pořadí 'a' a 'b'
:r3 bude soubor 's' obsahovat řádky v pořadí 'b' a 'a'
:r4 nebude soubor 's' existovat
:r1 ok protože při přesměrování výstupu se soubor vyprázdní dříve, než se provede příkaz
--
Po provedení příkazů<PRE>$ echo 'mravenec' > a; ls -l a
-rw-r--r-- 1 brandejs staff 9 dub 8 23:23 a
$ echo ferda >> a</PRE>bude mít soubor 'a' velikost
:r1 5
:r2 6
:r3 13
:r4 14
:r5 15
:r5 ok
--
Posloupnost příkazů<PRE>cat <<ukazka
ls -l ukazka
ukazka</PRE>předá na standardní výstup
:r1 obsah souboru 'ukazka', výstup příkazu 'ls -l ukazka', spustí se program 'ukazka' z běžného adresáře
:r2 text 'ls -l ukazka'
:r3 text 'ukazka'
:r4 nepředá se nic
:r5 předá se výstup programu 'ukazka'
:r2 ok
--
Spojení příkazů do kolony je
:r1 ls|grep txt|sort
:r2 ls;grep txt;sort
:r3 ls&grep txt&sort
:r4 ls&&grep txt&&sort
:r1 ok
--
Návratový kód seznamu<PRE>rm -rf adr;mkdir adr;touch adr/so;ls adr/so|sort|grep soubor</PRE>bude
:r1 0
:r2 nenulové kladné číslo
:r3 nenulové záporné číslo
:r4 prázdný řetězec
:r5 text s chybovým hlášením
:r2 ok
--
Návratový kód seznamu<PRE>rm -rf adr;touch adr/soubor;echo adr/soubor|sort|grep soubor</PRE>bude
:r1 0
:r2 nenulové kladné číslo
:r3 nenulové záporné číslo
:r4 prázdný řetězec
:r5 text s chybovým hlášením
:r1 ok
--
Příkaz spustíme na pozadí, když
:r1 za příkaz napíšeme &
:r2 před příkaz napíšeme &
:r3 za příkaz napíšeme $
:r4 před příkaz napíšeme $
:r1 ok
--
Pro příkaz spuštený na pozadí neplatí
:r1 lze ho ukončit speciálním znakem INTR
:r2 lze ho ukončit příkazem kill
:r3 standardní vstup se napojí na /dev/null
:r4 byl spuštěn pomocí oddělovače &
:r1 ok
--
Kam je napojen standardní vstup procesu spuštěného na pozadí?
:r1 /dev/null
:r2 /dev/tty
:r3 /dev/zero
:r4 /dev/console
:r1 ok
--
Který z příkazů předá na standardní výstup nejprve řádek obsahující 'b' a potom řádek obsahující 'a'?
:r1 sleep 2;echo a&sleep 1;echo b
:r2 sleep 1;echo a&sleep 2;echo b
:r3 (sleep 1;echo a)&sleep 2;echo b
:r4 (sleep 2;echo a)&sleep 1;echo b
:r4 ok
--
Spustím příkaz<PRE>(sleep 10;echo a)&sleep 10;echo b</PRE>a po spuštění stisknu zvláštní znak INTR (^C). Co se předá na standardní výstup dříve?
:r1 a
:r2 b
:r3 předají se řádky 'a' i 'b' ve stejný čas v náhodném pořadí
:r4 nepředá se nic
:r2 ok
--
Provedete posloupnost příkazů/operací:<PRE>cat > a&cat > b
ahoj
^D</PRE>Který soubor bude větší?
:r1 a
:r2 b
:r3 soubory 'a' i 'b' budou stejně velké a neprázdné
:r4 soubory 'a' i 'b' budou stejně velké a prázdné
:r5 jedno ^D nestačí, musí se stisknout dvakrát
:r2 ok
--
Jakým způsobem nelze úlohu dostat pod správu Job Controllu?
:r1 příkazem fg
:r2 příkazem bg
:r3 spuštěním s &
:r4 speciálním znakem SUSP (^Z)
:r1 ok
--
Úloha běžící na pozadí se přesune na popředí příkazem
:r1 lg
:r2 kill
:r3 nohup
:r4 fg
:r5 žádná jiná odpověď není správná
:r4 ok
--
Pro seznam neplatí
:r1 je to posloupnost žádneho nebo více příkazu
:r2 příkazy jsou odděleny novým řádkem nebo středníkem (;) nebo ampersandem (&)
:r3 shell provádí příkazy v pořadí, v jakém jsou zapsány
:r4 návratový kód seznamu je návratový kód prvního prováděného procesu
:r5 pokud příkaz končí ampersandem (&), ihned se zahájí provádení následujícího příkazu
:r4 ok
--
Oddělením dvou příkazů oddělovačem &&
:r1 spustíme první příkaz na popředí a druhý příkaz na pozadí
:r2 spustíme první příkaz na pozadí a druhý příkaz na popředí
:r3 provedeme druhý příkaz, jen když první příkaz vrátil nulový návratový kód
:r4 provedeme druhý příkaz, jen když první příkaz vrátil nenulový návratový kód
:r3 ok
--
K čemu slouží oddělovač && mezi dvěma příkazy?
:r1 druhý příkaz se provede, pokud je návratový kód prvního nenulový
:r2 druhý příkaz se provede, pokud je návratový kód prvního nula
:r3 druhý příkaz se spustí hned po spuštění prvního
:r4 druhý příkaz se spustí na pozadí hned po spuštění prvního
:r2 ok
--
Oddělením dvou příkazů oddělovačem ||
:r1 spustíme první příkaz na popředí a druhý příkaz na pozadí
:r2 spustíme první příkaz na pozadí a druhý příkaz na popředí
:r3 provedeme druhý příkaz, jen když první příkaz vrátil nulový návratový kód
:r4 provedeme druhý příkaz, jen když první příkaz vrátil nenulový návratový kód
:r4 ok
--
K čemu slouží oddělovač || mezi příkazy?
:r1 druhý příkaz se provede, pokud je návratový kód prvního nenulový
:r2 druhý příkaz se provede, pokud je návratový kód prvního nula
:r3 druhý příkaz se spustí hned po spuštění prvního
:r4 druhý příkaz se spustí na pozadí hned po spuštění prvního
:r1 ok
--
Příkazem<PRE>ls adresar 2>/dev/null || mkdir adresar</PRE>
:r1 vytvoříme adresář, pokud neexistuje
:r2 vytvoříme adresář, pokud příkaz ls vrátil nulový návratový kód
:r3 vypíšeme vždy obsah adresáře a vytvoříme nový adresář
:r4 vypíšeme obsah adresáře

:r1 ok
--
Vyberte nesprávné tvrzení o seznamu<PRE>cd zkusto&&rm *</PRE>
:r1 pokud adresář zkusto nebude existovat, zruší se všechny soubory běžného adresáře
:r2 první příkaz bude mít návratovou hodnotu 0, pokud existuje přístupný adresář zkusto
:r3 pokud adresář zkusto bude existovat, zruší se v něm všechny soubory
:r4 pokud první příkaz bude mít návratovou hodnotu 1, tak se druhý neprovede
:r1 ok
--
Příkaz<PRE>rm `cat files.txt`</PRE>
:r1 vymaže všechny soubory, které jsou uvedeny v souboru files.txt
:r2 vymaže soubor files.txt
:r3 vymaže soubor cat i soubor files.txt
:r4 vymaže soubor files.txt a také soubory, které jsou v něm uvedeny
:r1 ok
--
Příkaz<PRE>rm cat files.txt</PRE>
:r1 vymaže všechny soubory, které jsou uvedeny v souboru files.txt
:r2 vymaže soubor files.txt
:r3 vymaže soubor cat i soubor files.txt
:r4 vymaže soubor files.txt a také soubory, které jsou v něm uvedeny
:r3 ok
--
Příkaz<PRE>rm -rf *;touch a;echo 'ls' `ls`</PRE>předá na standardní výstup
:r1 ls a
:r2 a ls
:r3 obsah souboru 'a'
:r4 'ls' ls
:r5 ls `ls`
:r1 ok
--
Příkaz<PRE>echo a > seznam;rm 'seznam'</PRE>smaže soubor
:r1 a
:r2 seznam
:r3 příkaz je chybný
:r4 'a' i 'seznam'
:r2 ok
--
Příkaz<PRE>echo a > 'seznam';rm `cat seznam`</PRE>smaže soubor
:r1 a
:r2 seznam
:r3 příkaz je chybný
:r4 'a' i 'seznam'
:r1 ok
--
Příkaz<PRE>echo a > 'seznam';rm `echo seznam`</PRE>smaže soubor
:r1 a
:r2 seznam
:r3 příkaz je chybný
:r4 'a' i 'seznam'
:r2 ok
--
Příkaz<PRE>echo a > `seznam`;rm 'cat seznam'</PRE>smaže soubor
:r1 a
:r2 seznam
:r3 příkaz je chybný
:r4 'a' i 'seznam'
:r3 ok
--
Proměnná se v shellu "deklaruje" příkazem
:r1 var ADRESAR=/tmp
:r2 ADRESAR=/tmp
:r3 $ADRESAR=/tmp
:r4 strings $ADRESAR /tmp
:r2 ok
--
Jak vytvořím proměnou ADRESAR obsahující slovo EMPTY
:r1 ADRESAR=EMPTY
:r2 ADRESAR:=EMPTY
:r3 $ADRESAR=EMPTY
:r4 touch ADRESAR < EMPTY
:r1 ok
--
Zadali jsme<PRE>velikost=maxi</PRE>Jak vypíšeme řetězec "maxipes"?
:r1 echo $velikost."pes"
:r2 echo $[velikost]pes
:r3 echo $velikostpes
:r4 echo ${velikost}pes
:r4 ok
--
Deklarace proměnné spolu s jejím naplněním se provádí
:r1 jmeno_promenne=[hodnota]
:r2 jmeno_promenne:=[hodnota]
:r3 jmeno_promenne<[hodnota]
:r4 $jmeno_promenne:=[hodnota]
:r1 ok
--
Uživatelem zavedené proměnné se v shellu dědí do potomků
:r1 vždy
:r2 nikdy
:r3 jen proměnné označené příkazem export
:r4 jen proměnné označené příkazem import
:r5 jen proměnné označené příkazem child
:r3 ok
--
Jaké je nesprávné použití obsahu proměnné ADRESAR?
:r1 echo $ADRESAR/NEW
:r2 echo ${ADRESAR}NEW
:r3 echo $ADRESAR NEW
:r4 echo $ADRESARNEW
:r4 ok
--
Co provádí příkaz set spuštěný bez voleb a argumentů?
:r1 vyprázdní všechny proměnné
:r2 smaže všechny proměnné
:r3 vypíše všechny proměnné
:r4 exportuje všechny proměnné
:r3 ok
--
Ve kterém způsobu závorkování se provede příkaz ve vnořeném shellu?
:r1 (prikaz)
:r2 { prikaz;}
:r3 <prikaz;>
:r1 ok
--
Příkaz<PRE>cd `echo $OLDPWD`</PRE>se chová stejně jako:
:r1 pwd
:r2 cd ~
:r3 cd -
:r4 cd ..
:r3 ok
--
Který z příkazů má správnou syntaxi?
:r1 { echo ahoj;}
:r2 { echo ahoj }
:r3 {echo ahoj};
:r4 {echo ahoj}
:r5 {echo ahoj }
:r1 ok
--
Máme nastavený pracovní adresář /tmp/data. Které z použití příkazu <B>pwd</B> vypíše jiný pracovní adresář než /tmp/data?
:r1 cd $PWD;pwd
:r2 { cd $PWD;};pwd
:r3 { cd $HOME;};cd -;pwd
:r4 { cd $HOME;};pwd
:r5 (cd $HOME);pwd
:r4 ok
--
Co je uloženo v proměnné PWD
:r1 pracovní adresář
:r2 heslo, zašifrované jednosměrnou šifrou
:r3 domovský adresář
:r4 hesla všech uživatelů (zašifrovaná jednosměrnou šifrou)
:r1 ok
--
Mezi složené příkazy nepatří
:r1 for
:r2 case
:r3 while
:r4 until
:r5 find
:r5 ok
--
Co vykonává příkaz true?
:r1 vrací hodnotu true
:r2 vypisuje řetězec true dokud není ukončen
:r3 vrací návratový kód 0
:r4 vrací návratový kód 1
:r5 není to příkaz, je to logická hodnota
:r3 ok
--
[je
:r1 příkaz
:r2 otevírací závorka pro zápis podmínky
:r3 otevírací závorka pro zápis osmičkového čísla
:r4 symbol přesměrování
:r5 speciální znak
:r1 ok
--
Který z příkazů nesmaže všechny soubory z běžného adresáře?
:r1 for S in *; do rm $S; done
:r2 rm `ls`
:r3 rm `echo *`
:r4 for S in `ls`; do rm $S; done
:r5 všechny ostatní příkazy soubory smažou
:r5 ok
--
Kterým příkazem pošleme správně celý text e-mailem na zadanou adresu?
:r1 echo Milý Jeníčku,;echo posílám Ti seznam souborů;ls;echo;echo Tvůj Pepík>mail adresa@nekde
:r2 (echo Milý Jeníčku,;echo posílám Ti seznam souborů;ls;echo;echo Tvůj Pepík)>mail adresa@nekde
:r3 (echo Milý Jeníčku,;echo posílám Ti seznam souborů;ls;echo;echo Tvůj Pepík)|mail adresa@nekde
:r4 echo Milý Jeníčku,;echo posílám Ti seznam souborů;ls;echo;echo Tvůj Pepík|mail adresa@nekde
:r3 ok
--
Jaký příkaz můžeme použít, chceme-li na standardní výstup předat na řádku vždy jméno souboru, mezeru a znovu jméno souboru s příponou .o pro všechny soubory/položky běžného adresáře?
:r1 for JM in $PWD; do echo $JM ${JM}.o; done
:r2 for $JM in *; do echo $JM $JM.o; done
:r3 for JM in *; do echo $JM $JM.o; done
:r4 for JM in *; do echo JM JM.o; done
:r3 ok
--
Jaká je korektní syntaxe zápisu vyhodnocení podmínky?
:r1 if `ls soubor`; then rm soubor; fi
:r2 if 'ls soubor'; then rm soubor; fi
:r3 if ls soubor; then rm soubor; fi
:r4 if [ls soubor]; then rm soubor; fi
:r3 ok
--
Kterým příkazem nezjistím, zda položka 'adresar' je adresář?
:r1 if test -d adresar; then echo ok; fi
:r2 if [-d adresar]; then echo ok; fi
:r3 if ls -ld adresar|grep -q '^d'; then echo ok; fi
:r4 if pwd adresar; then echo ok; fi
:r4 ok
--
Který příkaz je syntakticky špatně?
:r1 if [$# -ge 3] && [$# -lt 6]; then echo ok; fi
:r2 if [$# -ge 3 && $# -lt 6]; then echo ok; fi
:r3 if [$# -ge 3 -a $# -lt 6]; then echo ok; fi
:r2 ok
--
Při kterém volání skriptu<PRE>#!/bin/bash
if [$# = 4 -a $1 != $2]; then echo ok; fi</PRE> se předá na standardní výstup 'ok'?
:r1 ./skript 'a' `echo a` a A
:r2 ./skript 1 2 '' 3 4
:r3 ./skript 1 2 '' 4
:r4 ./skript a \a b \b
:r3 ok
--
Který soubor nevypíše příkaz<PRE>ls x?[a-c] *</PRE>
:r1 žádný soubor, který by měl jméno delší než 4 znaky
:r2 soubor x1.c i když se v adresáři nachází
:r3 soubor xabc i když se v adresáři nachází
:r4 soubor x?abc* i když se v adresáři nachází

:r2 ok
--
Co předá na standardní výstup skript<PRE>#!/bin/bash
if [5 -lt 4]; then echo ano; else echo ne; fi</PRE>
 :r1 ano
 :r2 ne
 :r3 nepředá nic
 :r4 5
 :r5 4
:r3 ok
--
Jaká je správná syntaxe dotazu na existenci souboru?
 :r1 if [-f soubor] then echo ano fi
 :r2 if [-f soubor] then; echo ano fi;
 :r3 if [-f soubor]; then echo ano; fi
 :r4 if [-f soubor;]; then echo; ano; fi
 :r5 if [-f soubor;]; then echo ano; fi;
:r3 ok
--
Který příkaz je syntakticky správně a současně obsahuje co nejméně mezer?
 :r1 [-d adresar]&&echo ano
 :r2 [-d adresar]&&echo ano
 :r3 [-d adresar]&&echo ano
 :r4 [-d adresar] &&echo ano
 :r5 [-d adresar] && echo ano
:r3 ok
--
Pokud proměnná A obsahuje číslo větší než 5, který příkaz podmínku <TT>A > 5</TT> korektně vyhodnotí a předá na standardní výstup 'ano'?
 :r1 if [$A > 5] then echo ano fi;
 :r2 if [$A > 5]; then echo ano; fi
 :r3 if [$A -gt 5] then echo ano; fi;
 :r4 if [$A -gt 5]; then echo ano; fi
:r4 ok
--
Kterým/i znakem/znaky začínají v shellu komentáře?
 :r1 #
 :r2 //
 :r3 !
 :r4 %
 :r5 /*
:r1 ok
--
Kterým příkazem spustím skript v běžném shellu?
 :r1 /skript
 :r2 .skript
 :r3 . skript
 :r4 ! skript
 :r5 žádná jiná odpověď není správná
:r3 ok
--
Chci-li spustit skript (např. příkazem ./skript), musím mít na soubor se skriptem právo
 :r1 spuštění
 :r2 čtení a spuštění
 :r3 čtení, zápis a spuštění
 :r4 čtení
 :r5 zápis a spuštění
:r2 ok
--
Mějme skript<PRE>#!/bin/sh
echo $4$1$3$2</PRE>Co předá na standardní výstup příkaz <TT>./skript a "" b f</TT>
 :r1 f a b
 :r2 fab
 :r3 af b
 :r4 afb
 :r5 b a f
:r2 ok
--
Co předá na standardní výstup příkaz "echo $2", jestliže byl předtím proveden příkaz "set -- a b c"?
 :r1 --
 :r2 a
 :r3 b
 :r4 c
:r3 ok
--
Co předá na standardní výstup příkaz<PRE>(exit 1; exit 0); echo $?</PRE>
 :r1 nic, shell se ukončí
 :r2 0
 :r3 1
 :r4 ?
:r3 ok
--
Pracovní soubor v adresáři /tmp s unikátním jménem (žádné jiný aktivní proces spuštěný z téhož skriptu nepoužije stejné jméno) vytvořím příkazem
 :r1 touch /tmp/pomocny
 :r2 touch /tmp/pomocny.$?
 :r3 touch /tmp/pomocny.$$
 :r4 touch /tmp/pomocny.$%
 :r5 touch /tmp/pomocny.$!
:r3 ok
--
Příkaz<PRE>ls ~</PRE>předá na standardní výstup položky z
 :r1 běžného adresář
 :r2 domovského adresáře
 :r3 kořenového adresáře
 :r4 adresáře /tmp
 :r5 adresáře /bin
:r2 ok
--
V běžném adresáři máte položky
<TT>a
aa
amanda
am da</TT> (mezera ve jméně)
<TT>AMANDA</TT>
Která jména předá na standardní výstup příkaz <TT>ls a*a</TT>
 :r1 aa amanda am da
 :r2 aa amanda
 :r3 amanda
 :r4 aa amanda am da AMANDA
 :r5 a aa amanda am da AMANDA
:r1 ok
--
V běžném adresáři máte položky
<TT>a
aa
amanda
am da</TT> (mezera ve jméně)
<TT>AMANDA</TT>
Která jména předá na standardní výstup příkaz <TT>echo [a-m][!0-9]</TT>
 :r1 a
 :r2 aa
 :r3 amanda   am da
 :r4 amanda am da AMANDA
 :r5 [a-m][!0-9]
:r2 ok